Top Answers to Python Interview Questions

1. Compare between Java and Python.

Criteria	Java	Python
Ease of use	Good	Excellent
Speed of coding	Average	Excellent
Data types	Statically typed	Dynamically typed
Data Science and Machine Learning applications	Average	Excellent

2. What is Python?

Python is a high-level, interpreted, interactive, and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently, whereas the other languages use punctuation, and it has fewer syntactical constructions than the other languages.

3. What are the key features of Python?

- Python is an interpreted language, so it doesn’t need to be compiled before execution, unlike languages such as C.
- Python is dynamically typed, so there is no need to declare a variable with the data type. Python Interpreter will identify the data type on the basis of the value of the variable.

For example, in Python, the following code line will run without any error:

```
a = 100
a = "Intellipaat"
```

- Python follows an object-oriented programming paradigm with the exception of having access specifiers. Other than access specifiers (public and private keywords), Python has classes, inheritance, and all other usual OOPs concepts.
- Python is a cross-platform language, i.e., a Python program written on a Windows system will also run on a Linux system with little or no modifications at all.
- Python is literally a general-purpose language, i.e., Python finds its way in various domains such as web application development, automation, Data Science, Machine Learning, and more.

4. What is the purpose of PYTHONPATH environment variable?

PYTHONPATH has a role similar to PATH. This variable tells Python Interpreter where to locate the module files imported into a program. It should include Python source library directory and the directories containing Python source code. PYTHONPATH is sometimes preset by Python Installer.

5. What is the purpose of PYTHONSTARTUP, PYTHONCASEOK, and PYTHONHOME environment variables?

- **PYTHONSTARTUP**: It contains the path of an initialization file having Python source code. It is executed every time we start the interpreter. It is named as .pythonrc.py in Unix, and it contains commands that load utilities or modify PYTHONPATH.
- **PYTHONCASEOK**: It is used in Windows to instruct Python to find the first case-insensitive match in an import statement. We can set this variable with any value to activate it.
- **PYTHONHOME**: It is an alternative module search path. It is usually embedded in PYTHONSTARTUP or PYTHONPATH directories to make switching of module libraries easy

6. Which data types are supported in Python?

Python has six standard data types:

- Numbers
- Strings
- Tuples
- Lists
- Set
- Dictionaries

7. What is the difference between lists and tuples?

Lists	**Tuples**
Lists are mutable, i.e., they can be edited.	Tuples are immutable (they are lists that cannot be edited).
Lists are usually slower than tuples.	Tuples are faster than lists.
Syntax: list_1 = [10, ‘Srinivas’, 20]	**Syntax**: tup_1 = (10, ‘Srinivas’ , 20)

8. How is memory managed in Python?

- Memory in Python is managed by Python private heap space. All Python objects and data structures are located in a private heap. This private heap is taken care of by Python Interpreter itself, and a programmer doesn’t have access to this private heap.
- Python memory manager takes care of the allocation of Python private heap space.
- Memory for Python private heap space is made available by Python’s in-built garbage collector, which recycles and frees up all the unused memory.

9. Explain Inheritance in Python with an example.

As Python follows an object-oriented programming paradigm, classes in Python have the ability to inherit the properties of another class. This process is known as inheritance. Inheritance provides the code reusability feature. The class that is being inherited is called a **superclass** and the class that inherits the superclass is called a **derived** or **child class**. Following types of inheritance are supported in Python:

- **Single inheritance**: When a class inherits only one superclass
- **Multiple inheritance**: When a class inherits multiple superclasses
- **Multilevel inheritance**: When a class inherits a superclass and then another class inherits this derived class forming a ‘parent, child, and grandchild’ class structure
- **Hierarchical inheritance**: When one superclass is inherited by multiple derived classes

10. What is a dictionary in Python?

Python dictionary is one of the supported data types in Python. It is an unordered collection of elements. The elements in dictionaries are stored as key–value pairs. Dictionaries are indexed by keys.

For example, below we have a dictionary named ‘dict’. It contains two keys, Country and Capital, along with their corresponding values, India and New Delhi.

```
dict={‘Country’:’India’,’Capital’:’New Delhi’, }
```

11. Can you write an efficient code to count the number of capital letters in a file?

The normal solution for this problem statement would be as follows:

```
with open(SOME_LARGE_FILE) as countletter:
     count = 0
     text = countletter.read()
     for character in text:
          if character.isupper():
               count += 1

print(count)
```

To make this code more efficient, the whole code block can be converted into a one-liner code using the feature called generator expression. With this, the equivalent code line of the above code block would be as follows:

```
count sum(1 for line in countletter for character in line if character.isupper())
```

12. Write a code to sort a numerical list in Python.

The following code can be used to sort a numerical list in Python:

```
list = [“2”, “5”, “7”, “8”, “1”]
list = [int(i) for i in list]
list.sort()
print(list)
```

13. How will you reverse a list in Python?

The function **list.reverse()** reverses the objects of a list.

14. How will you remove the last object from a list in Python?

```
list.pop(obj=list[-1]):
```

Here, **−1** represents the last element of the list. Hence, the **pop()** function removes the last object (**obj**) from the list.

15. What are negative indexes and why are they used?

To access an element from ordered sequences, we simply use the index of the element, which is the position number of that particular element. The index usually starts from 0, i.e., the first element has index 0, the second has 1, and so on.
When we use the index to access elements from the end of a list, it’s called reverse indexing. In reverse indexing, the indexing of elements starts from the last element with the index number ‘−1’. The second last element has index ‘−2’, and so on. These indexes used in reverse indexing are called negative indexes.

16. What are split(), sub(), and subn() methods in Python?

These methods belong to Python RegEx ‘re’ module and are used to modify strings.

- **split()**: This method is used to split a given string into a list.
- **sub()**: This method is used to find a substring where a regex pattern matches, and then it replaces the matched substring with a different string.
- **subn()**: This method is similar to the sub() method, but it returns the new string, along with the number of replacements.
- **17. How are range and xrange different from one another?**

Functions in Python, range() and xrange() are used to iterate in a for loop for a fixed number of times. Functionality-wise, both these functions are the same. The difference comes when talking about Python version support for these functions and their return values.

The range() Method	**The xrange() Method**
In Python 3, xrange() is not supported; instead, the range() function is used to iterate in for loops.	The xrange() function is used in Python 2 to iterate in for loops.
It returns a list.	It returns a generator object as it doesn’t really generate a static list at the run time.
It takes more memory as it keeps the entire list of iterating numbers in memory.	It takes less memory as it keeps only one number at a time in memory.

18. Define pickling and unpickling in Python.

Pickling is the process of converting Python objects, such as lists, dicts, etc., into a character stream. This is done using a module named ‘pickle’, hence the name pickling.
The process of retrieving the original Python objects from the stored string representation, which is the reverse of the pickling process, is called **unpickling**.

19. What is a map function in Python?

The **map()** function in Python has two parameters, function and iterable. The map() function takes a function as an argument and then applies that function to all the elements of an iterable, passed to it as another argument. It returns an object list of results.

For example:

```
def calculateSq(n):
        return n*n
numbers = (2, 3, 4, 5)
result = map( calculateSq, numbers)
print(result)
```

20. Write a code to get the indices of N maximum values from a NumPy array.

We can get the indices of N maximum values from a NumPy array using the below code:

```
import numpy as np
ar = np.array([1, 3, 2, 4, 5, 6])
print(ar.argsort()[-3:][::-1])
```

21. What is a Python module?

Modules are independent Python scripts with the .py extension that can be reused in other Python codes or scripts using the import statement. A module can consist of functions, classes, and variables, or some runnable code. Modules not only help in keeping Python codes organized but also in making codes less complex and more efficient. The syntax to import modules in Python is as follows:

```
import module_name   # include this code line on top of the script
```

22. What do file-related modules in Python do? Can you name some file-related modules in Python?

Python comes with some file-related modules that have functions to manipulate text files and binary files in a file system. These modules can be used to create text or binary files, update their content, copy, delete, and more.

Some file-related modules are os, os.path, and shutil.os. The os.path module has functions to access the file system, while the shutil.os module can be used to copy or delete files.

23. Explain the use of the 'with' statement and its syntax.

In Python, using the ‘with’ statement, we can open a file and close it as soon as the block of code, where ‘with’ is used, exits. In this way, we can opt for not using the close() method.

```
with open("filename", "mode") as file_var:
```

24. Explain all file processing modes supported in Python.

Python has various file processing modes.

- For opening files, there are three modes:
    - read-only mode (r)
    - write-only mode (w)
    - read–write mode (rw)

- For opening a text file using the above modes, we will have to append ‘t’ with them as follows:
  - read-only mode (rt)
  - write-only mode (wt)
  - read–write mode (rwt)
- Similarly, a binary file can be opened by appending ‘b’ with them as follows:
  - read-only mode (rb)
  - write-only mode (wb)
  - read–write mode (rwb)
- To append the content in the files, we can use the append mode (a):
  - For text files, the mode would be ‘at’
  - For binary files, it would be ‘ab’

25. Is indentation optional in Python?

Indentation in Python is compulsory and is part of its syntax.

All programming languages have some way of defining the scope and extent of the block of codes; in Python, it is indentation. Indentation provides better readability to the code, which is probably why Python has made it compulsory.

26. How are Python arrays and Python lists different from each other?

In Python, when we say ‘arrays’, we are usually referring to ‘lists’. It is because lists are fundamental to Python just as arrays are fundamental to most of the low-level languages.

However, there is indeed a module named ‘array’ in Python, which is used or mentioned very rarely. Following are some of the differences between Python arrays and Python lists.

Arrays	Lists
Arrays can only store homogeneous data (data of the same type).	Lists can store heterogeneous and arbitrary data.
Since only one type of data can be stored, arrays use memory for only one type of objects. Thus, mostly, arrays use lesser memory than lists.	Lists can store data of multiple data types and thus require more memory than arrays.
The length of an array is pre-fixed while creating it, so more elements cannot be added.	Since the length of a list is not fixed, appending items to it is possible.

27. Write a code to display the contents of a file in reverse.

To display the contents of a file in reverse, the following code can be used:

```
for line in reversed(list(open(filename.txt))):
print(line.rstrip())
```

28. Differentiate between NumPy and SciPy.

NumPy	SciPy
NumPy stands for Numerical Python.	SciPy stands for Scientific Python.
It is used for efficient and general numeric computations on numerical data saved in arrays. E.g., sorting, indexing, reshaping, and more.	This module is a collection of tools in Python used to perform operations such as integration, differentiation, and more.
There are some linear algebraic functions available in this module, but they are not full-fledged.	Full-fledged algebraic functions are available in SciPy for algebraic computations.

29. Which of the following is an invalid statement?

1. xyz = 1,000,000
2. x y z = 1000 2000 3000
3. x,y,z = 1000, 2000, 3000
4. x_y_z = 1,000,000

Answer: 2

30. Can we make multiline comments in Python?

Python does not have a specific syntax for including multiline comments like other programming languages. However, programmers can use triple-quoted strings (docstrings) for making multiline comments as when a docstring is not used as the first statement inside a method, it gets ignored by Python parser.

31. What would be the output if I run the following code block?

```
list1 = [2, 33, 222, 14, 25]
print(list1[-2])
```

1. 14
2. 33
3. 25
4. Error

Answer: 14

32. Write a command to open the file c:\hello.txt for writing.

```
f= open("hello.txt", "wt")
```

33. What is __init__ in Python?

Equivalent to constructors in OOP terminology, __init__ is a reserved method in Python classes. The __init__ method is called automatically whenever a new object is initiated. This method allocates memory to the new object as soon as it is created. This method can also be used to initialize variables.

34. What do you understand by Tkinter?

Tkinter is an in-built Python module that is used to create GUI applications. It is Python's standard toolkit for GUI development. Tkinter comes with Python, so there is no installation needed. We can start using it by importing it in our script.

35. Is Python fully object oriented?

Python does follow an object-oriented programming paradigm and has all the basic OOPs concepts such as inheritance, polymorphism, and more, with the exception of access specifiers. Python doesn't support strong encapsulation (adding a private keyword before data members). Although, it has a convention that can be used for data hiding, i.e., prefixing a data member with two underscores.

36. What is lambda function in Python?

A lambda function is an anonymous function (a function that does not have a name) in Python. To define anonymous functions, we use the 'lambda' keyword instead of the 'def' keyword, hence the name 'lambda function'. Lambda functions can have any number of arguments but only one statement.

37. What is self-keyword in Python?

Self-keyword is used as the first parameter of a function inside a class that represents the instance of the class. The object or the instance of the class is automatically passed to the method that it belongs to and is received in the 'self-keyword.' Users can use another name for the first parameter of the function that catches the object of the class, but it is recommended to use 'self-keyword' as it is more of a Python convention.

38. What are control flow statements in Python?

Control flow statements are used to manipulate or change the execution flow of a program. Generally, the flow of the execution of a program runs from top to bottom, but certain statements (control flow statements) in Python can break this top-to-bottom order of execution. Control flow statements include decision-making, looping, and more.

39. What is the difference between append() and extend() methods?

Both append() and extend() methods are methods used to add elements at the end of a list.

- **append(element)**: Adds the given element at the end of the list that called this append() method
- **extend(another-list)**: Adds the elements of another list at the end of the list that called this extend() method

40. What are loop interruption statements in Python?

There are two types of loop interruption statements in Python that let users terminate a loop iteration prematurely, i.e., not letting the loop run its full iterations.

Following are the two types of loop interruption statements:

- **Python break statement**: This statement immediately terminates the loop entirely, and the control flow of the program is shifted directly to the outside of the loop.
- **Python continue statement**: Continue statement terminates the current loop iteration and moves the control flow of the program to the next iteration of the loop, letting the user skip only the current iteration.

41. What is docstring in Python?

Python lets users include a description (or quick notes) for their methods using documentation strings or docstrings. Docstrings are different from regular comments in Python as, rather than being completely ignored by the Python Interpreter like in the case of comments, Python documentation strings can actually be accessed at the run time using the dot operator when docstring is the first statement in a method or function.

Are you interested in learning Python from experts? Enroll in our online* Python Course in Bangalore *today!

42. What is the output of the following?

```
x = ['ab', 'cd']
print(len(list(map(list, x))))
```

Output:

```
[['a', 'b'], ['c', 'd']].
```

Explanation: Each element of x is converted into a list.

43. Which one of the following is not the correct syntax for creating a set in Python?

1. set([[1,2],[3,4],[4,5]])
2. set([1,2,2,3,4,5])
3. {1,2,3,4}
4. set((1,2,3,4))

Answer: set([[1,2],[3,4],[4,5]])

Explanation: The argument given for the set must be an iterable.

44. What is functional programming? Does Python follow a functional programming style? If yes, list a few methods to implement functionally oriented programming in Python.

Functional programming is a coding style where the main source of logic in a program comes from functions.

Incorporating functional programming in our codes means writing pure functions.

Pure functions are functions that cause little or no changes outside the scope of the function. These changes are referred to as side effects. To reduce side effects, pure functions are used, which makes the code easy-to-follow, test, or debug.

Python does follow a functional programming style. Following are some examples of functional programming in Python.

- **filter()**: Filter lets us filter some values based on a conditional logic.

```
>>> list(filter(lambda x:x>6,range(9))) [7, 8]
```

- **map()**: Map applies a function to every element in an iterable.

```
>>> list(map(lambda x:x**2,range(5))) [0, 1, 4, 9, 16, 25]
```

- **reduce()**: Reduce repeatedly reduces a sequence pair-wise until it reaches a single value.

```
>>> from functools import reduce >>> reduce(lambda x,y:x-y,[1,2,3,4,5]) -13
```

45. How does Python Flask handle database requests?

Flask supports a database-powered application (RDBS). Such a system requires creating a schema, which needs piping the schema.sql file into the sqlite3 command. So, we need to install the sqlite3 command in order to create or initiate the database in Flask.

Flask allows to request for a database in three ways:

- **before_request()**: They are called before a request and pass no arguments.
- **after_request()**: They are called after a request and pass the response that will be sent to the client.
- **teardown_request()**: They are called in a situation when an exception is raised and responses are not guaranteed. They are called after the response has been constructed. They are not allowed to modify the request, and their values are ignored.

46. Write a Python program to check whether a given string is a palindrome or not, without using an iterative method. Note: A palindrome is a word, phrase, or sequence that reads the same backward as forward, e.g., madam, nurses run, etc.

```
def fun(string):
s1 = string
s = string[::-1]
if(s1 == s):
return true
else:
return false
print(fun(“madam”))
```

47. Write a Python program to calculate the sum of a list of numbers.

```
def sum(num):
if len(num) == 1:
return num[0]               #with only one element in the list, sum result will be equal to the el
ement.
else:
return num[0] + sum(num[1:])
print(sum([2, 4, 5, 6, 7]))
```

Output:

```
24
```

48. Do we need to declare variables with data types in Python?

No. Python is a dynamically typed language, I.E., Python Interpreter automatically identifies the data type of a variable based on the type of value assigned to the variable.

49. How will you read a random line in a file?

We can read a random line in a file using the random module.

For example:

```
import random
def read_random(fname):
    lines = open(fname).read().splitlines()
    return random.choice(lines)
    print(read_random ('hello.txt'))
```

50. Write a Python program to count the total number of lines in a text file.

```
def file_count(fname):
    with open(fname) as f:
        for i, 1 in enumerate(f):
        pass
    return i+1
print("Total number of lines in the text file: ", file_count("file.txt"))
```